Evaluation of SCTP Multistreaming over Satellite Links [1]

Mohammed Atiquzzaman

School of Computer Science

University of Oklahoma, Norman, OK 73019-6151

Email: atiq@ieee.org, Tel: (405) 325 8077, Fax: (405) 325 4044.

William Ivancic

NASA Glenn Research Center

21000 Brookpark Rd. MS 54-8,

Cleveland, OH 44135.

Abstract

The Stream Control Transmission Protocol has been developed as a reliable transport PSTN signaling messages over an IP network. Although developed based on the concepts of TCP, multistreaming is one of its powerful features to overcome some of the bottlenecks of byte stream oriented TCP. In this paper, we study the impact of multistreaming to increase the performance of SCTP over a error prone satellite network. We first show that multistreaming results in higher goodput than single streams when the receiver buffer is constrained as in the case of wireless portable handheld devices. We then demonstrate that multistreaming feature of SCTP results in reduced buffer requirements at the receiver in the presence of losses in the satellite network.

1 Introduction

The classical PSTN/ISDN networks use separate networks for voice and signaling. SS7 signaling messages are usually carried over an expensive dedicated network infrastructure. To enable transfer of signaling traffic over IP networks [1, 2], a family of protocols [3] is being developed based on SCTP [4], a new end to end message-based reliable transport protocol. SCTP provides an important element for the convergence of voice and data networks. The family of protocols allow interworking between SS7 network elements and IP-based elements. SCTP removes some of the restrictions of TCP, such as strict byte ordered delivery which is too limiting in the case of signaling where mutually independent transactions may be included within each packet; TCP's strict byte order of delivery in such cases is not only unnecessary but also reduces the throughput by introducing

[1]This work was supported by NASA grant no. NAG3-2528

head of line blocking. Although initially conceived as a reliable transport protocol to carry PSTN signaling messages over IP, it has emerged into a highly reliable and fault-tolerant transport protocol which has the potential to replace TCP in the future.

SCTP is particularly effective for applications that require the following features [5]:

- framing of reliable data streams,
- ordered transport of data but can transfer multiple message sequences that are unrelated,
- transfer of messages that hold no particular sequence or relationship to one another or can be correlated and sequenced at the application level,
- network layer redundancy

An additional advantage of SCTP over TCP is SCTP's ability to deliver all data completely unordered yet still reliably. This can be advantageous in applications which deal with a large number of independent transactions.

Multistreaming and *multihoming* are the two major differences between TCP and SCTP. Multihoming allows two end points to set up an association with multiple IP addresses for each end point. One of the addresses is designated as the primary while the rest can be used as backup in the case of failure of the primary address, or when the upper layer application explicitly requests the use of the backup. Retransmission of lost packets can also be done over the secondary address.

Multistreaming allows data from a number of upper layer applications to be multiplexed onto one channel (called association in SCTP) as shown in Fig. 2. Congestion control is applied to the association instead of individual streams. Sequencing of data is done within a stream; if a packet belonging to a certain stream is lost, packets (from that stream) following the lost one will be stored in the receiver's stream buffer until the lost packet is retransmitted from the source. However, data from other streams can still be passed to upper layer applications. This avoids the head of line blocking found in TCP where only one stream carries data from all the different upper layer applications.

The head of line blocking at the individual streams in shown in Fig. 1 with four applications corresponding to the four streams. Packets are are identified by Stream Sequence Numbers (SSN) [4] which are unique within a stream. SSN 11 has been delivered to application number 1 and SSN 12 is arriving at the buffer of stream 1. SSN 9 for the second stream is lost; SSNs 10, 11, 12 are therefore queued in the buffer of the second stream. Arriving SSN 13 will also be queued. Similarly

SSN 3 of stream 3 is missing. For application 4, SSN 21 is being delivered to the application while arriving SSN 23 will be queued in the buffer because of missing SSN 22. This illustrates the fact that packets arriving on stream 1 can still be delivered to the upper layer application although stream 2 and 3 are (and stream 4 will be) blocked because of lost packets.

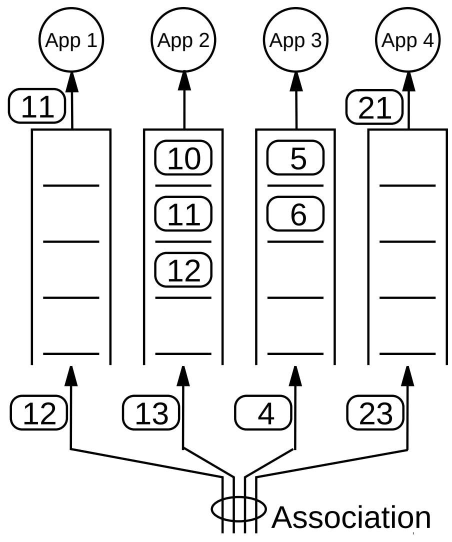


Figure 1: Illustration showing head of line blocking of individual streams at the receiver.

It is anticipated that multistreaming will allow web like applications, where multiple objects (such as images) are typically downloaded concurrently, to run faster than using TCP. The inherent multistreaming capability of SCTP will also speed up access to image databases [6] and transmission of multimedia traffic [7]. It will even speedup the downloading of a single image when it is compressed using multiresolution schemes (such as EZW or progressive JPEG) as has been demonstrated in [8].

The HTTP protocol uses TCP as the transport mechanism. Early HTTP protocol used a separate TCP connection to fetch each URL, increasing the load on HTTP servers and causing congestion on the Internet [9, 10]. The use of inline images and other associated data often requires a client to make multiple requests of the same server in a short amount of time. The HTTP/1.1 protocol requires implementing persistent connections [11] where the same TCP connection is used for many URL fetches (see Sec. 8.1 of [12]). Future HTTP protocols could use the multistreaming feature of SCTP to allow fast downloading of web pages by simultaneously downloading different objects over different streams belonging to the *same association*.

The amount of data that can be sent by a TCP (or SCTP) sender depends on the congestion

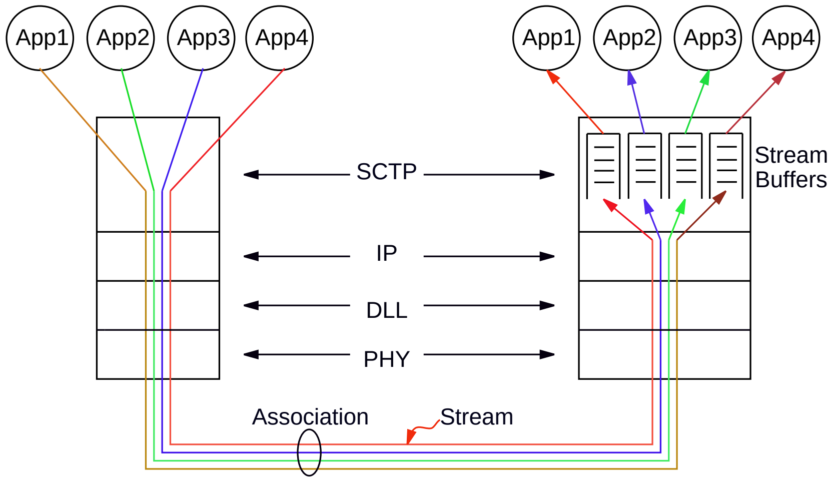


Figure 2: An SCTP association consisting of four streams carrying data from four upper layer applications.

window and the receiver buffer size. When a packet is lost in the network, subsequent packets are queued up in the receiver buffer for resequencing until the lost packet is retransmitted from the source and arrives at the receiver. In the case of limited receiver buffer size, the receiver could run out of buffer space due to head of line blocking resulting in the sender being unable to send data. In the case of multistreaming, some of the streams could be delivering packets to the upper layer when a particular stream in blocked while waiting for a lost packet to arrive. Multistreaming could therefore result in higher throughput and reduced buffer requirements than single streaming.

The *objective* of this paper is to study the performance of SCTP in a satellite network which is characterized by errors and long propagation delays. In particular, we would like to investigate the effect of multistreaming on the goodput of SCTP and the buffer requirements at the receiver which is connected using a satellite link. The *contributions* of this work are as follows:

- Determine the goodput of SCTP over satellite networks for various link error rates and receiver buffer sizes.

- Demonstrate that multistreaming can significantly reduce the receiver buffer size requirements which would be helpful in designing wireless handheld devices.

- Determined the optimal buffer requirement at the receiver which can be used to dimension the receiver buffer size.

The rest of the paper is organized as follows. Differences between TCP and SCTP congestion control are discussed in Sec. 2 followed by definitions and notations used in this paper in Sec. 3. The simulation topology and the assumptions used are described in Sec. 4 followed by results in Sec. 5. Finally, summary of our findings and conclusions of this work are presented in Sec. 6.

2 Congestion control of TCP and SCTP

2.1 TCP Congestion control

TCP congestion control is driven by loss of packets which are interpreted by TCP as congestion in the network. TCP throttles back its transmission rate in response to packet loss. Packet loss can be detected by timeout or duplicate acknowledgements from the receiver. Slow start is invoked when packet loss is detected by timeout, and duplicate acknowledgement results in fast retransmission.

2.1.1 Slow Start and Congestion Avoidance

During slow start [13], packet transmission rate from the source is doubled every Round Trip Time (RTT) until *cwnd* reaches *ssthresh* at which point the source enters congestion avoidance during which the source rate increases linearly until a packet loss is detected or the receiver runs out of buffer space. If allowed by the congestion window and the receiver window, the number of segments transmitted increases by one for every RTT during the congestion avoidance phase.

2.1.2 Fast Retransmit

When a packet loss is detected by three consecutive DUP ACKs, the Fast Retransmit algorithm [13] retransmits the lost data without waiting for timeout.

2.1.3 SACK

Cumulative acknowledgement in the ACKs sent from the receiver only provides information about the lowest numbered packet which is lost. This means that only one packet can be recovered per RTT; multiple packet losses within the same RTT can not be recovered by DUP ACKs. To solve this problem, SACK [14] can be used to determine all packets within an RTT which are missing. Fast Retransmit can then be used to recover these multiple packets lost within an RTT.

2.2 SCTP Congestion control

SCTP congestion control is based on the well proven rate-adaptive window-based congestion control scheme of TCP. This ensures fairness for both protocols as they work together in the Internet. SCTP provides reliable transmission and detects lost, reordered, duplicated or corrupt packets. It provides reliability by retransmitting lost or corrupt packets. In the next section, we describe the differences between TCP and SCTP congestion control mechanisms.

2.3 Differences between TCP and SCTP congestion control

The congestion control of SCTP is based on schemes similar to those of TCP such as slow start, congestion avoidance, etc. Some of the differences between the congestion control mechanisms of TCP and SCTP which are relevant for this paper are given below. A more detailed list can be found in [15].

- **SACK:** SACK is optional in TCP; it considers the information carried in SACK as advisory information only. SACK is mandatory in SCTP; however, it considers the information in the Gap Ack Blocks in the SACK chunk as advisory.

- **cwnd:** In the case of non-SACK TCP, *cwnd* represents the upper bound between the highest acknowledged sequence number and the largest DATA chunk that can be sent within the congestion window. In SCTP, cwnd (in bytes) controls the amount of outstanding data.

  Although cwnd can only be increased by the advancement of Cumulative TSN Ack Point by the receipt of a SACK, new data can be clocked out on the receipt of Dup ACK. This is because cwnd represents the amount of data in flight, and receipt of Dup ACK represents packets that have left the network and have been delivered to the destination.

  The increase of cwnd in SCTP depends directly on the number of bytes ACKed rather than the number of ACKs received as in TCP.

- **Fast Recovery:** SCTP does not require explicit Fast Recovery (as may be used in some versions of TCP). Without even increasing the "cwnd", SCTP can clock out new data when Dup ACKs are received. This helps SCTP to maintain throughput in the presence of packet loss in the network.

- Gap Ack Blocks: An unlimited number of Gap Ack Blocks are allowed in SCTP SACK. TCP

SACK limits the number to at most four because of the limitation on the size of the option field.

- Message Boundaries: Data transported over TCP between two end points is a stream or sequence of bytes; user message boundaries are not preserved at the destination. For SCTP, user message boundaries are preserved at the destination.

3 Definitions and Notations

We define the following terms which will be used to describe the results in the following sections:

- *Goodput* is defined as the total number of packets (without considering the retransmitted packets) reaching the destination during the simulation period. In our experiments, we measure the goodput by the highest numbered TSN reaching the destination during the simulation time.

- *Optimal receiver buffer size* is the smallest amount of receiver buffer for which the SCTP goodput is independent of the receiver buffer size.

We use the following notations in the rest of the paper.

- s = number of streams per SCTP association.
- r_i = transmission rate of link i in Mbps.
- d_i = propagation delay of link i in msec.
- ϵ_i = error rate of link i. It is the probability that a packet is lost in the link due to errors.
- B = Receiver buffer size in bytes.
- a_rwnd = Advertised receiver window size in bytes.
- $cwnd$ = Congestion window size in bytes.

4 Simulation Setup and Assumptions

We have used simulation to evaluate the performance of SCTP multistreaming over a satellite link. The simulation was carried out using the *ns* simulator [16] with an SCTP patch from the

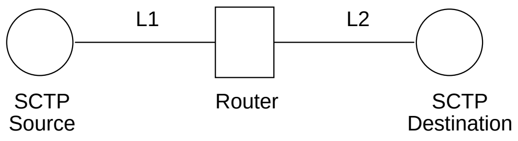


Figure 3: Network simulation topology.

University of Delaware. The network topology for the simulation is shown in Figure 3 where an SCTP source sends one way traffic to an SCTP sink through a router, possibly located in the satellite. $L_i, 1 \leq i \leq 2$ are links whose transmission rate, propagation delay and random error rate are expressed by the tuple (r_i, d_i, ϵ_i). In our simulation, we assume that L1 and L2 are terrestrial and satellite links respectively.

We make the following assumptions regarding the simulation setup:

- Data transfers are long. The SCTP source has an infinite supply of data which is being transferred to the destination using *ftp*.
- The terrestrial link is error free, i.e. $\epsilon_1 = 0$.
- Packets are of fixed length equal to one MTU.
- The upper layer at the destination is always ready to accept data.

5 Results

In this section, we first present the results for the performance of SCTP in the absence and presence of errors. We then show the improvement of SCTP goodput due to multistreaming for various error rates and receiver buffer sizes. Finally, we determine the optimal receiver buffer size for different error rates and number of streams. Throughout this study, we have used $r_1 = 5$, $r_2 = 10$, $d_1 = d_2 = 130$, and $\epsilon_1 = 0$. The values of r_1 and r_2 were chosen to avoid any congestion loss in the network, thereby leaving all possible losses due to satellite link errors.

5.1 Effect of Zero Errors

Fig. 4 shows SCTP goodput as a function of the receiver buffer size for $s = 4$ and $\epsilon_2 = 0$ representing zero packet loss due to errors. In the absence of packet losses, there is no blocking at the destination. *cwnd* initially increases and then becomes constant when it reaches B. After this point, the source

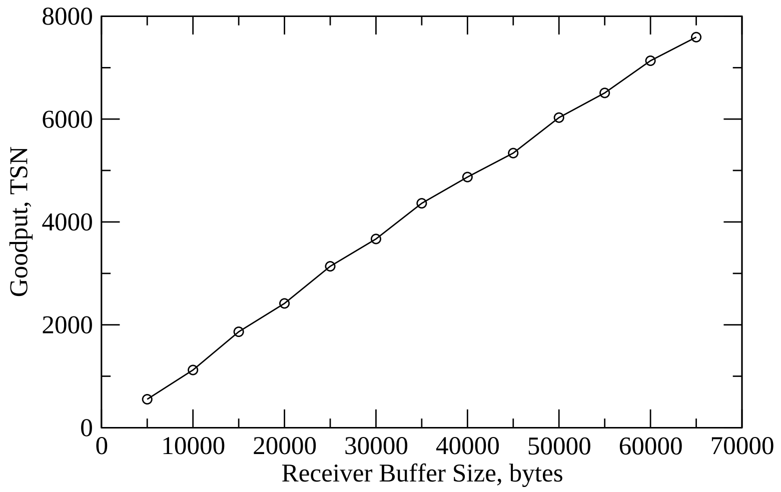


Figure 4: Goodput as a function of receiver buffer size for $\epsilon_2 = 0$ and $s = 4$.

sends a burst of $\left\lfloor \frac{B}{MTU} \right\rfloor$ packets every RTT as seen in Fig. 5 which plots the TSNs of the packets leaving and Cumulative TSN of the ACKs arriving at the source as a function of time. The goodput therefore, depends directly on the receiver buffer size as seen in Fig. 4. The y-axis shows (TSN *mod* 100).

In the absence of packet losses, there is no blocking at the destination buffer. The goodput is therefore independent of the number of streams, i.e. although Fig. 4 is shown for $s = 4$, it applies for any value of s.

5.2 Effect of Errors

Fig. 6 shows the goodput as a function of the receiver buffer size for $s = 4$ and $\epsilon_2 = 0.01$, 0.03 and 0.05. For a particular error rate, the goodput initially increases as B increases indicating that the goodput is constrained by the receiver buffer size. This is evident from Figure 7 which shows *cwnd* and *a_rwnd* for $s = 4$, $\epsilon_2 = 0.01$ and $B = 15K$. Receiver buffer size of 15K was chosen to make the goodput of the connection constrained by the receiver buffer size as seen by frequent dropping of *a_rwnd* below one MTU which in turn restricts the increase of *cwnd* beyond 15K.

As B increases (Fig. 6), a point is reached after which any further increase of B does not have any effect on the goodput; at this point, the goodput is limited by the congestion control mechanism of SCTP invoked by packet losses due to link errors. Further increase in goodput can only be achieved by lowering the link error rate as seen in Fig. 6. The fact that the goodput is limited by *cwnd* for large values of B is evident from Fig. 8 which shows *cwnd* and *a_rwnd* for $s = 4$,

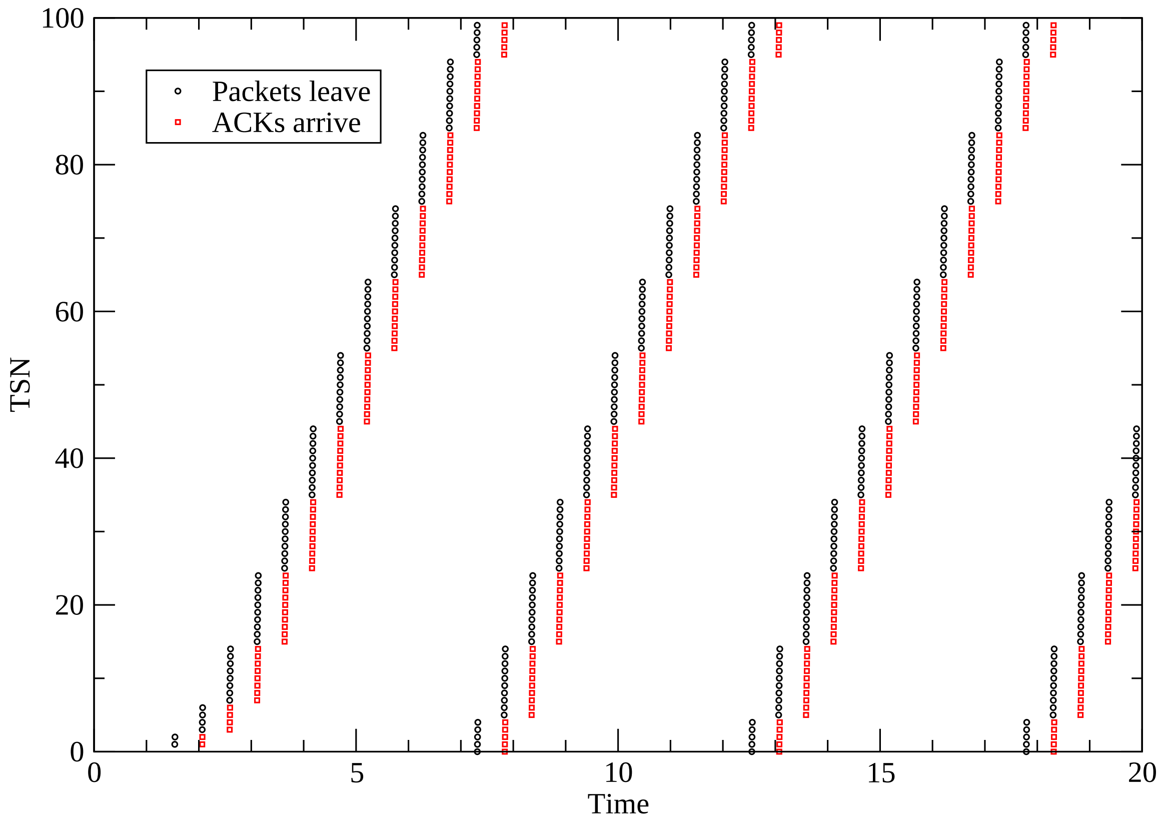


Figure 5: Packets sent and acknowledgements received at the source for $\epsilon_2 = 0$, $s = 4$ and $B = 15K$.

$\epsilon_2 = 0.01$ and $B = 35K$. a_rwnd falls below one MTU only once, while at other times, the decrease of $cwnd$ (which is a measure of goodput) is governed by SCTP's congestion control mechanisms (slow start, congestion avoidance, fast retransmit, etc.) after packet drops. Our observation that the goodput at the flat region of the curves in Fig. 6 is limited by the congestion control mechanism of SCTP is validated by the fact that a_rwnd never fell below the MTU in the flat region of the goodput curve; i.e. the receiver buffer size was not a limiting factor in the goodput.

Fig. 9 shows packets sent and ACKs received at the source. We can see long delays in retransmitting lost packets while waiting for DUP ACKs resulting from the lost packets. These long delays and the drop of $cwnd$ results in reduced goodput when receiver buffer is not a constraint.

5.3 Effect of Multistreaming

We have shown in the previous sections that the receiver buffer size could be a limiting factor in the case of small buffers. In this section, we show how multistreaming *reduces the receiver buffer requirements* and *increases the goodput* by avoiding HOL blocking at the receiver.

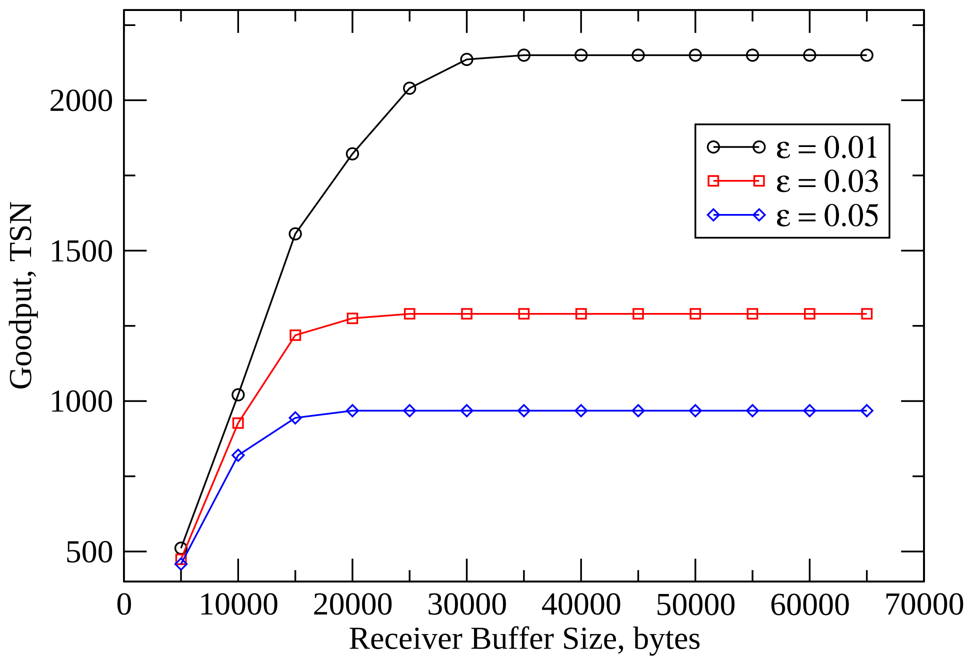


Figure 6: Goodput as function of receiver buffer size for different error rates and $s = 4$.

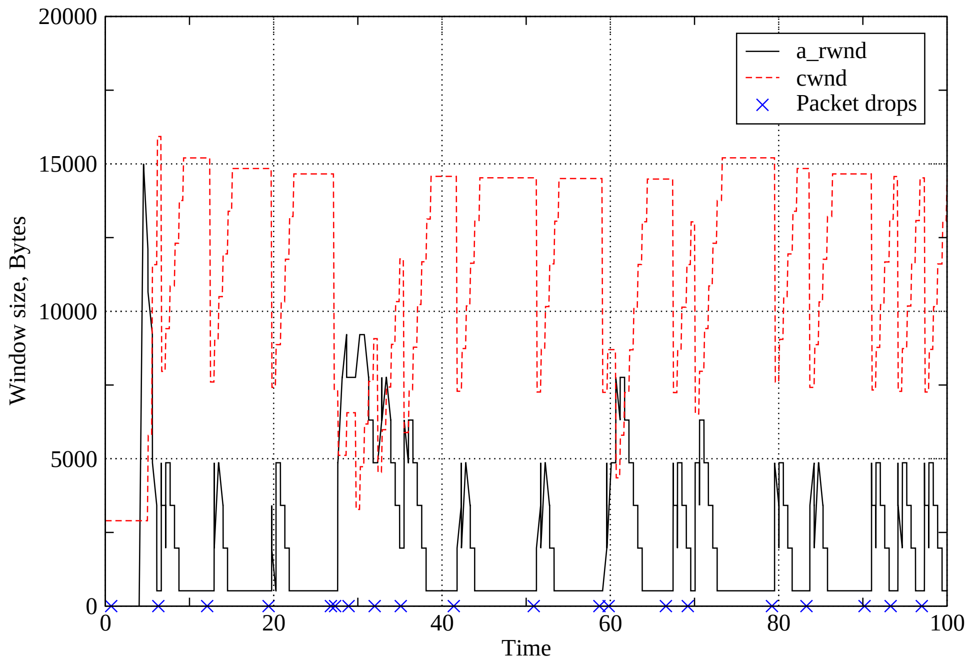


Figure 7: Advertised receiver window and congestion window with $B = 15K$, $\epsilon_2 = 0.01$, and $s = 4$.

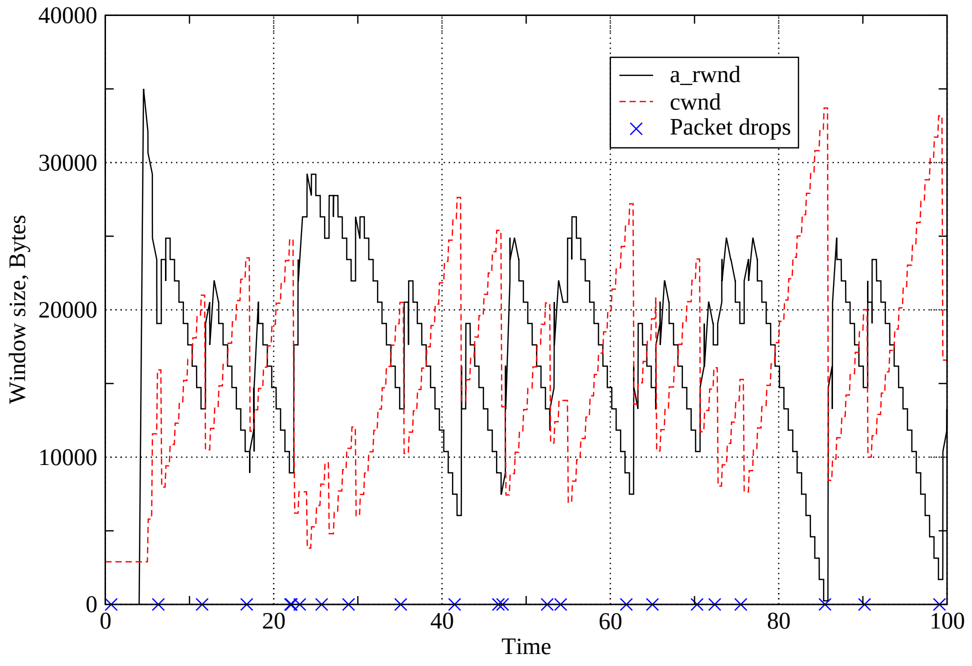


Figure 8: Advertised receiver window and congestion window with $B = 35K$, $\epsilon_2 = 0.01$, and $s = 4$.

Figure 10 shows the goodput of an SCTP connection using one and four streams for error rates of 1% and 5% in the bottleneck link. The goodput of SCTP with four streams per connection is better than one stream per connection for small receiver buffer sizes. This is because, for small receiver buffer sizes and a single-stream per association, the sender is often prevented from sending packets because of lack of receiver buffer space arising due to HOL blocking. However, in the case of multiple streams per association, some of the streams may be able to pass packets to the upper layers even though other streams are blocked due to lost packets belonging to those streams. Fewer packets are dropped when the error rate is low. This results in higher goodput due to a smaller chance of blocking at the receiver.

Figure 11 compares the goodput of SCTP for one and four streams as a function of the error rate for $B = 15K$. The higher goodput of multistreaming is shown for various error rates. As an example, four streams per association results in about 10% increase in thruput over one stream at 3% error rate. A small receiver buffer size (15K) was chosen to demonstrate the *advantage of multistreaming when the buffer size is limited.*

The advertised receiver window for one and four streams is shown in Fig. 12 for $\epsilon_2 = 5\%$ and $B = 15K$. It is seen that, for a single stream, *a_rwnd* frequently falls below 1500 bytes (one MTU) thereby restricting the sender from sending packets. In the case of four streams, the advertised

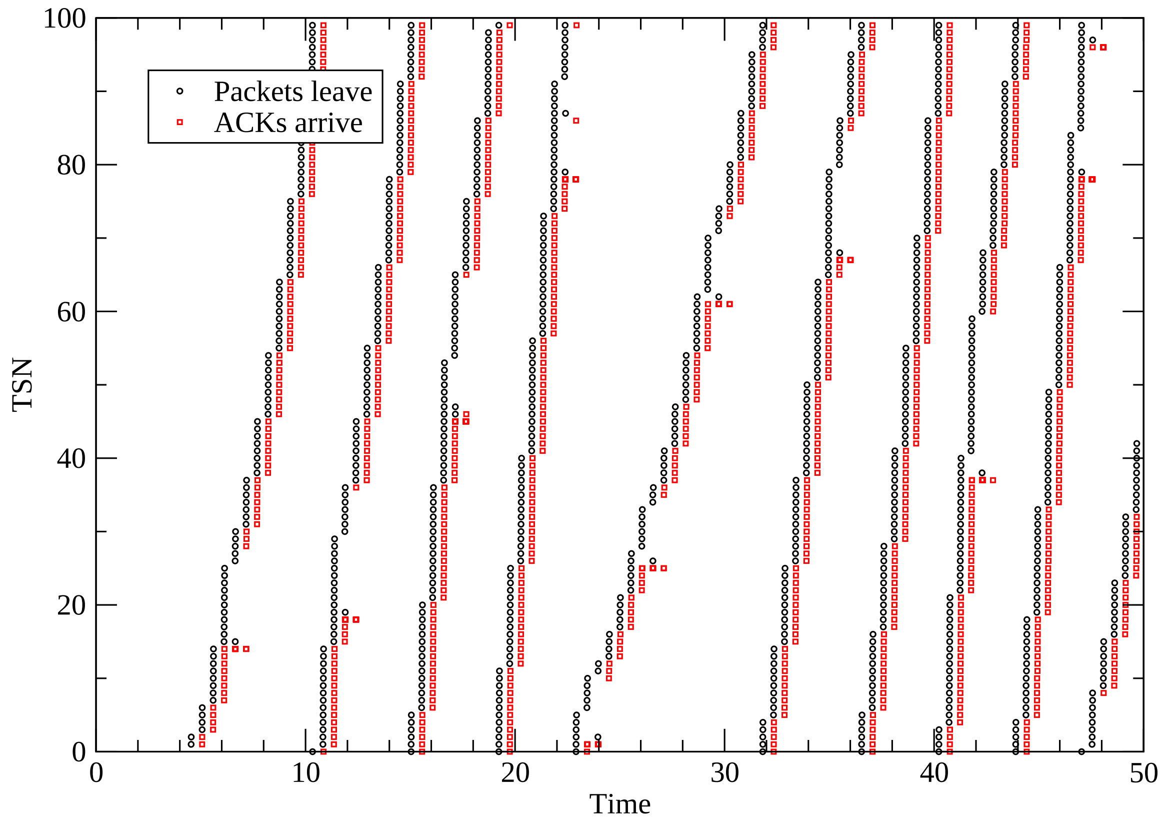


Figure 9: Packets sent and acknowledgements received at the source for $\epsilon_2 = 0.01$, $s = 4$ and $B = 35K$.

receiver buffer size falling below one MTU is less frequent than the case of a single stream. This results in higher goodput for the case of multistreaming. Note that one MTU is the minimum size required by the sender to send data.

5.4 Optimal Receiver Buffer Size

It has been shown in Sec. 5.2 that as the receiver buffer size is increased, the goodput of SCTP in the presence of errors (see Fig. 6) becomes independent of the receiver buffer size. In this section, we determine the *optimal receiver buffer size* (see Sec. 3 for definition) for various error rates. Receiver buffer size larger than the optimal receiver buffer size does not contribute to increasing the goodput, and hence is wasted. The optimal receiver buffer size can be used to dimension the buffer size at the receiver for portable mobile devices (such as PDAs and next generation wireless phones) where it is desirable to reduce the memory size because of weight and power restrictions.

Figure 13 shows the optimal receiver buffer size as a function of the error rate for one and four

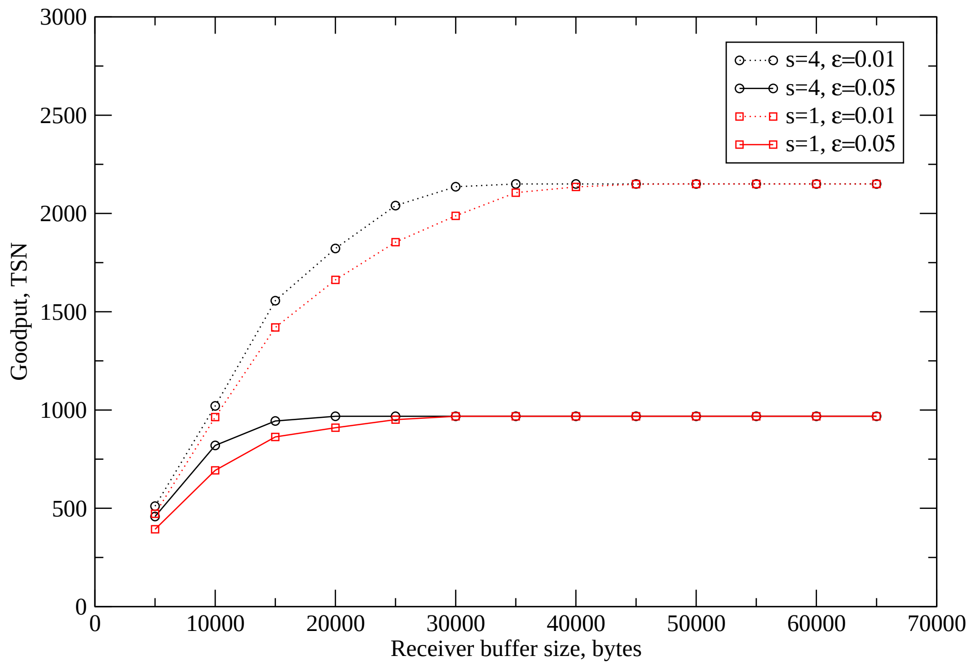


Figure 10: Goodput as function of receiver buffer size for various wireless error rates with one and four streams.

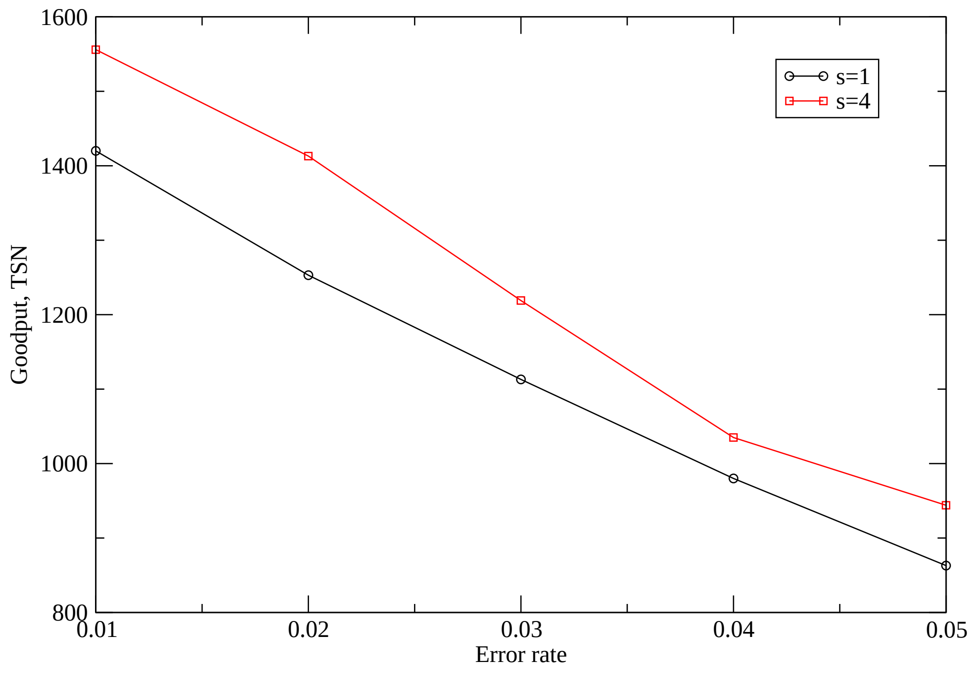
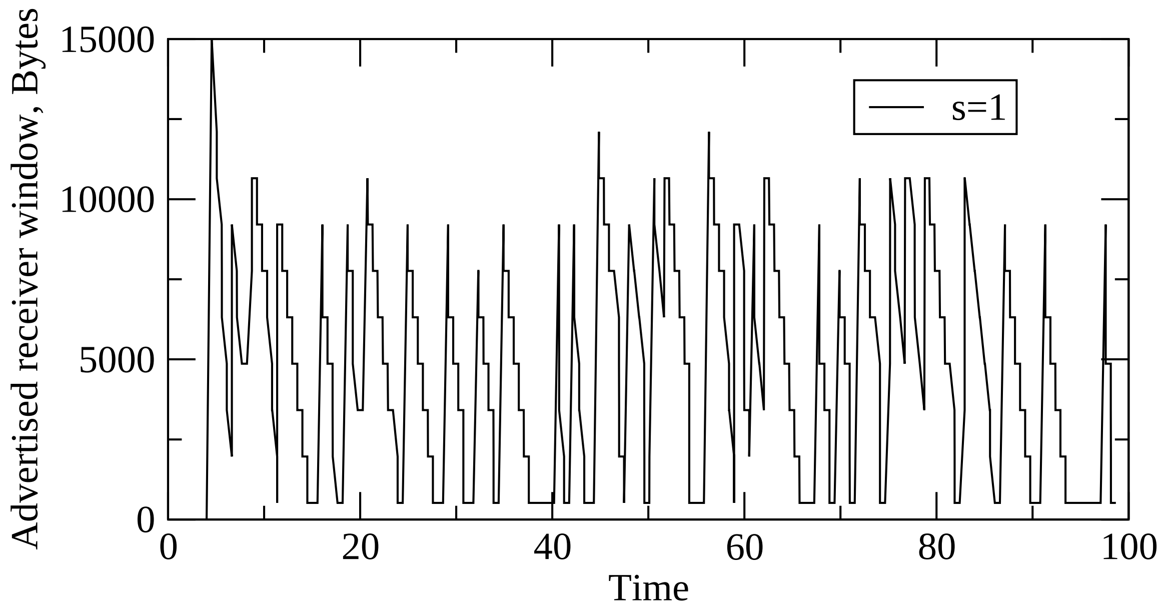


Figure 11: Goodput as function of error for one and four streams for $B = 15K$.



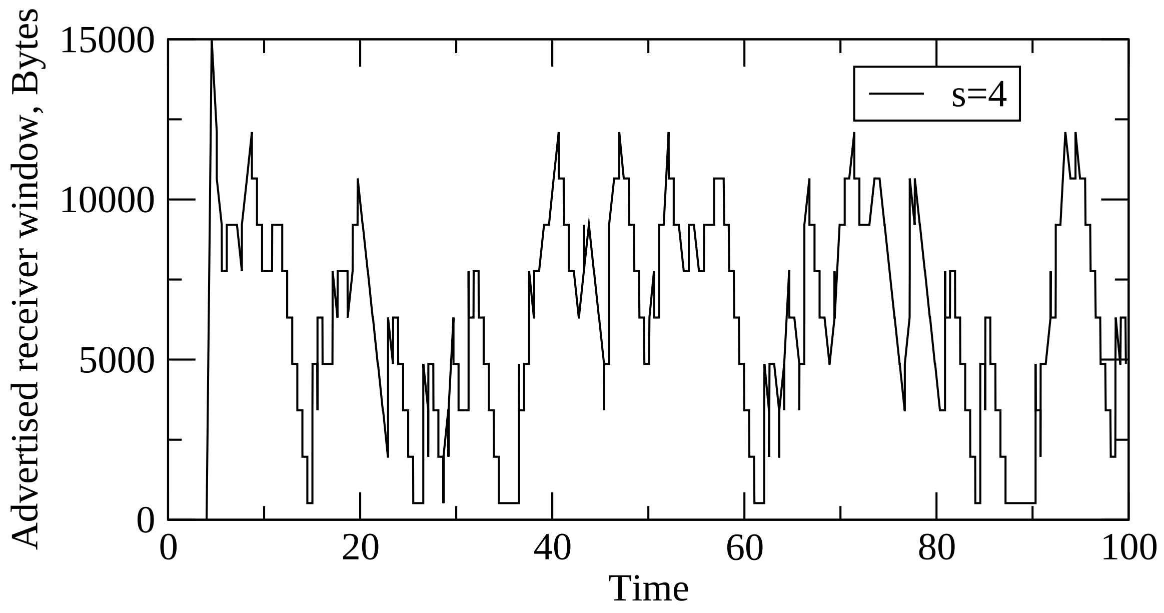


Figure 12: Advertised receiver window for one and four streams. $\epsilon_2 = 0.05$, $B = 15K$.

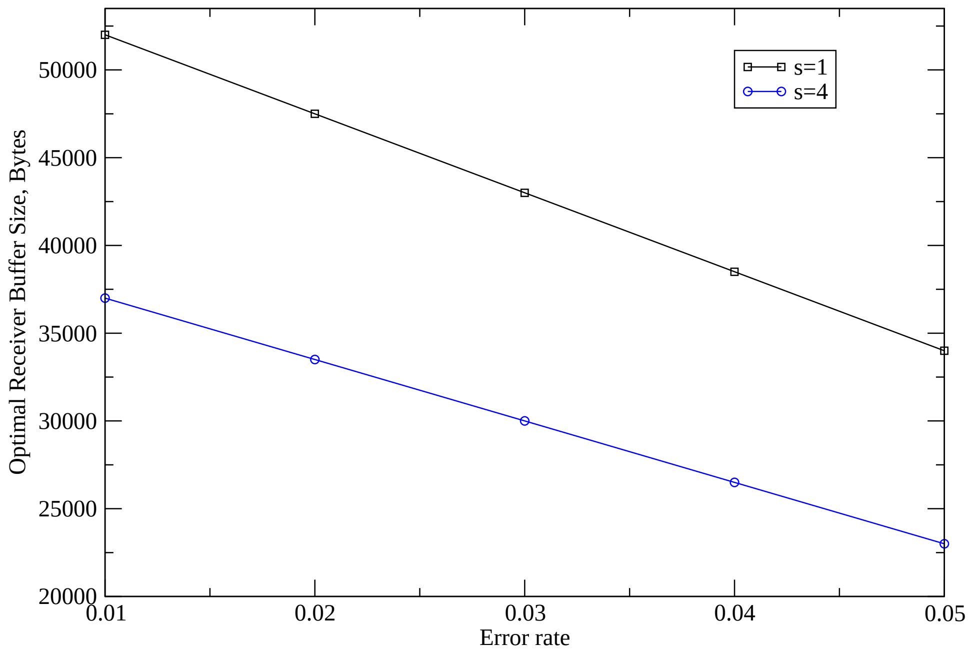


Figure 13: Optimal receiver buffer size vs. error rate for one and four streams.

streams. The buffer size required for multistreaming is significantly lower than for single streams. For example, four streams result in a saving of about 35% of receiver buffer space over a single stream for 1% error rate. Similar saving in also evident for other error rates. The optimal receiver buffer size was determined as the receiver buffer size for which the goodput becomes constant (i.e. independent of the receiver buffer size) in Fig. 6.

Figure 13 also shows that the optimal receiver buffer size is smaller for higher error rates. This is consistent with our observation in Sec. 5.2 that *as the error rate increases, the goodput is increasingly dominated by the congestion control mechanism, and to a lesser extent by the receiver buffer size.*

6 Conclusions

Multistreaming allows multiple applications to transmit data over an association using streams. We have shown that multistreaming improves the goodput of SCTP for the case of limited receiver buffer size in the presence of errors in a satellite link. With the proliferation of handheld devices operating over wireless (or satellite/radio) links having limited receiver buffer capacity, SCTP will perform better than TCP (which uses a single stream) for such devices.

The results presented in this paper also demonstrate that multistreaming can reduce the buffer requirements at the reciver, and hence would be attractive for wireless handheld devices where the reduction of weight and power requirements is of outmost importance.

References

[1] B. Doorselaer and J. Bouwen, "Evolving the telephone network towards IP technology," *Journal of the British Telecommunication Engineers*, vol. 1, no. 3, pp. 126–131, Jul/Sep 2000.

[2] M. Hassan, A. Nayandoro, and M. Atiquzzaman, "Internet telephony: Services, technical challenges and products," *IEEE Communications Magazine*, vol. 38, no. 4, pp. 96–103, April 2000.

[3] L. Ong, I. Rytina, M. Garcia, and H. Schwarzbauer Et.Al., "Framework architecture for signaling transport." RFC 2719, Oct 1999.

[4] R. Stewart, Q. Xie, K. Morneault, and C. Sharp et. al., "Stream control transmission protocol." RFC 2960, Oct 2000.

[5] T. Ravier, R. Brennan, and T. Curran, "Experimental studies of SCTP multihoming," *First Joint IEI/IEE Symposium on Telecommunications Systems Research*, Dublin, Ireland, p. ****, Nov 27, 2001.

[6] E.A. Heredia, C. Teng, and M. Ozkan, "Using multiresolution and multistreaming for faster access in image database broadcast," *1998 International Conference on Image Processing*, Chicago, IL, pp. 784–788, Oct 4-7 1998.

[7] R. Simon, T. Znati, and R. Sclabassi, "XCAP: a multistream routing algorithm for multimedia traffic," *Third IEEE International Conference on Multimedia Computing and Systems*, Hiroshima, Japan, pp. 104–107, June 17-23 1996.

[8] P.T. Conrad and G.J. Heinz, "SCTP in battlefield networks," *MILCOM 2001*, Washington, DC, p. ***, Oct 2001.

[9] Joe Touch, John Heidemann, and Katia Obraczka, "Analysis of HTTP performance." http://www.isi.edu/lsam/publications/http-perf/index.html, August 1996.

[10] S. Spero, "Analysis of HTTP performance problems." http://www.ibiblio.org/mdma-release/http-prob.html.

[11] J. Heidemann, K. Obraczka, and J. Touch, "Modeling the performance of HTTP over several transport protocols," *IEEE/ACM Transactions on Networking*, vol. 5, no. 5, pp. 616–630, Oct 1997.

[12] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee, "Hypertext transfer protocol – HTTP/1.1." RFC 2068, Jan 1997.

[13] M. Allman, V. Paxon, and W. Stevens, "TCP congestion control." RFC 2581, April 1999.

[14] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgement options." RFC 2018, Oct 1996.

[15] R. Brennan and T. Curran, "SCTP congestion control: Initial simulation studies," *International Teletraffic Congress (ITC 17)*, Brazil, p. ***, 2001.

[16] "ns-2 network simulator." http://www.isi.edu/nsnam/ns/.